Bloque III: El nivel de transporte

Tema 7: Transferencia fiable de datos

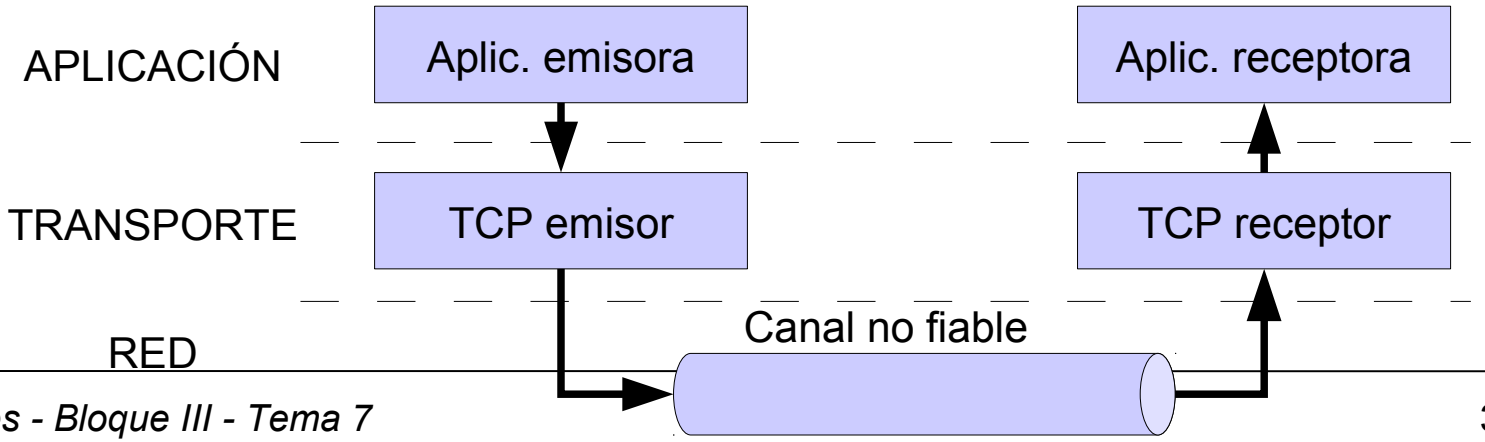
Índice

- Bloque III: El nivel de transporte
  - Tema 7: Transferencia fiable de datos
    - Introducción
    - Protocolos ARQ de parada y espera
    - Protocolos ARQ retroceder N
    - Protocolos ARQ de repetición selectiva

- **Lecturas recomendadas:**
  - Capítulo 3 sección 3.4 “Redes de Computadores: Un enfoque descendente”. James F. Kurose, Keith W. Ross. Addison Wesley.

Introducción

- Servicio de transferencia fiable de datos:
  - Los datos son recibidos correctamente (no se corrompen).
  - Los datos se reciben completos (no se pierde nada).
  - Los datos se entregan en el orden en el que fueron enviados.
- Problema: implementar un servicio de transferencia fiable de datos sobre un servicio (protocolo) de transferencia de datos **no fiable**.
  - Por ejemplo, TCP sobre IP.
  - Aunque este problema aparece en otros niveles: enlace y aplicación.

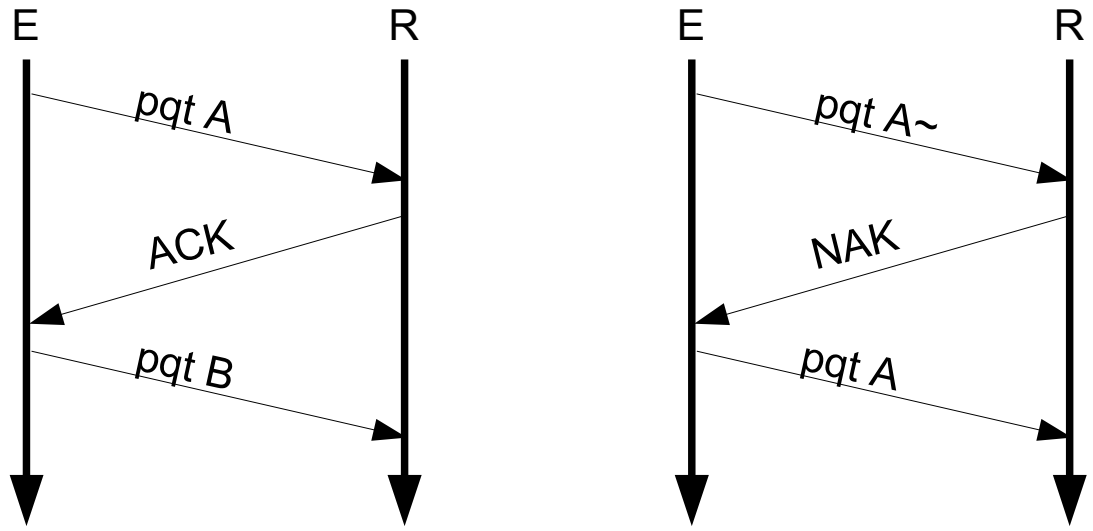


Introducción

- Si el canal es fiable → No hay nada que hacer
- Si el canal puede corromper los datos (pero mantiene el orden y no pierde datos) → Es necesario confirmar que los datos se han recibido correctamente (o no).
  - ACK: confirmación positiva.
  - NAK: confirmación negativa.
- Este tipo de protocolos se denominan ARQ (**Automatic Repeat reQuest**) y se basan en:
  - Detección de errores (en el receptor): **checksum**.
  - Realimentación del receptor: confirmando los datos recibidos con los **ACK** y NAK.
  - **Retransmisión**: por parte del emisor, si un paquete no se ha recibido correctamente.

Protocolos ARQ de parada y espera

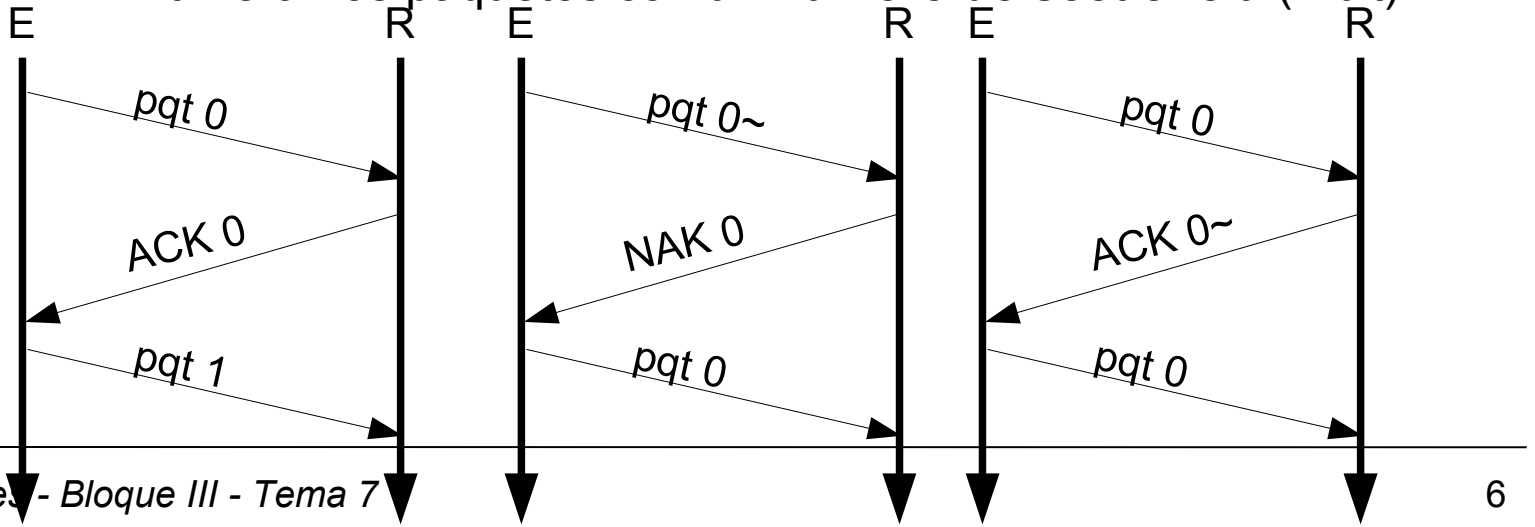
- El emisor envía un paquete y espera a recibir confirmación.
  - Si es un ACK → Envía un nuevo paquete de datos.
  - Si es un NAK → Retransmite el paquete.
- El receptor recibe un paquete y envía la confirmación.



- **Protocolo de parada y espera**: el emisor no envía datos nuevos hasta confirmar que el receptor ha recibido correctamente los datos anteriores.

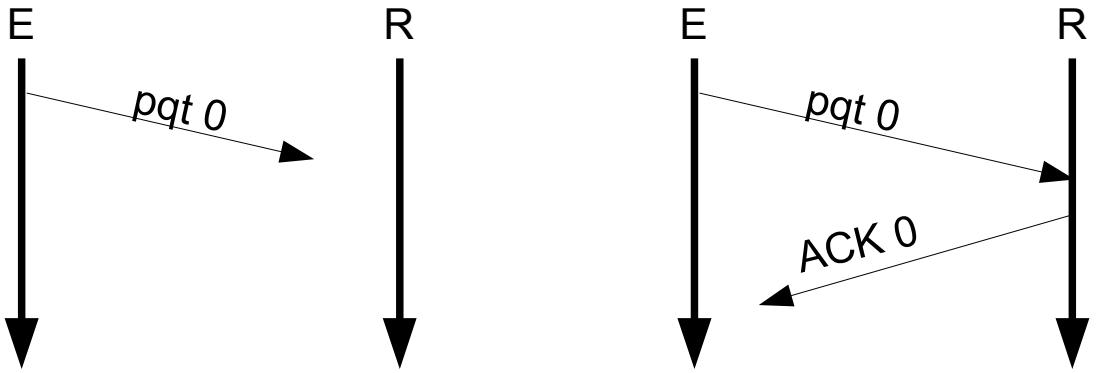
Protocolos ARQ de parada y espera

- Pero, ¿qué pasa si se corrompe un ACK o NAK?
  - El emisor detecta que está corrompido.
  - Pero no sabe si el receptor ha recibido correctamente los datos.
- Soluciones:
  - Incluir un mensaje nuevo: CACK (Corrupted ACK) pero, ¿qué pasa si se corrompe?
  - Código corrector de errores: soluciona este problema, pero no nos sirve en caso de pérdida.
  - Retransmitir: puede generar **paquetes duplicados**. El receptor debe ser capaz de distinguir cuando son datos nuevos o repetidos → Numerar los paquetes con un **número de secuencia** (1 bit).



Protocolos ARQ de parada y espera

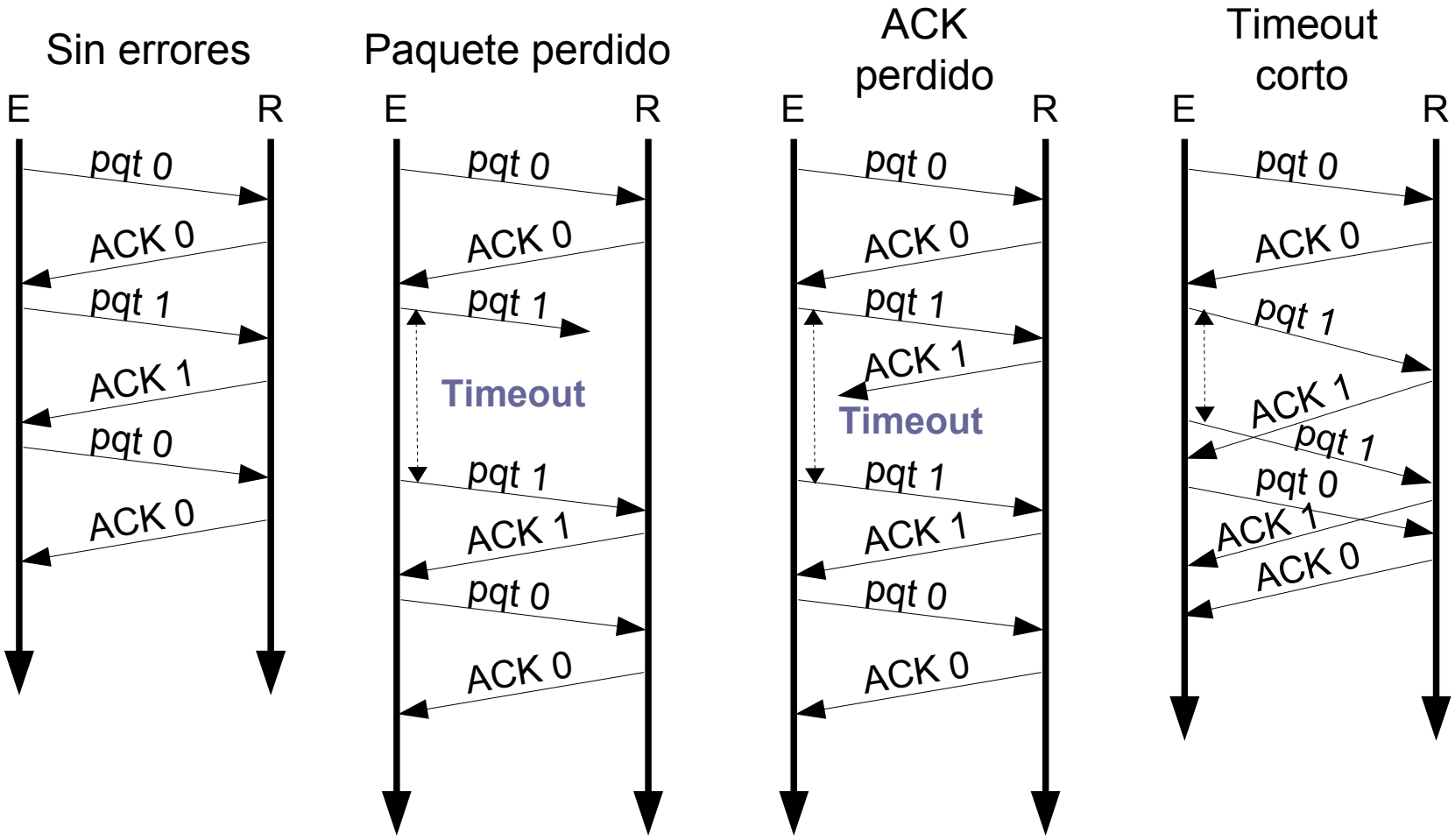
- ¿Y si el canal también puede perder datos?



- Solución: esperar y retransmitir.
- Esperar sí, pero ¿cuánto tiempo?
  - Idealmente: el tiempo de ida y vuelta (RTT – Round Trip Time) de un paquete en la red + un tiempo extra de procesamiento.
  - En muchas redes (p.e. Internet) el RTT no tiene un valor máximo.
  - Si espero demasiado → Tardo mucho en recuperarme del error.
  - Si espero poco → Envío muchos datos repetidos, para nada.
- Requiere incorporar un **temporizador** (o timeout) en el emisor para cada paquete de datos enviado.

Protocolos ARQ de parada y espera

- Como los números de secuencia de los paquetes se alternan entre 0 y 1, se suele denominar protocolo de bit alternante.



Protocolos ARQ de parada y espera

- El protocolo anterior nos garantiza la fiabilidad, pero ¿cuál es su rendimiento?

Tamaño pqt 0: 1000 bytes

Velocidad de transmisión: 1Gbps

Retardo de transmisión: 1000 * 8 / 1Gbps = 0.008 ms

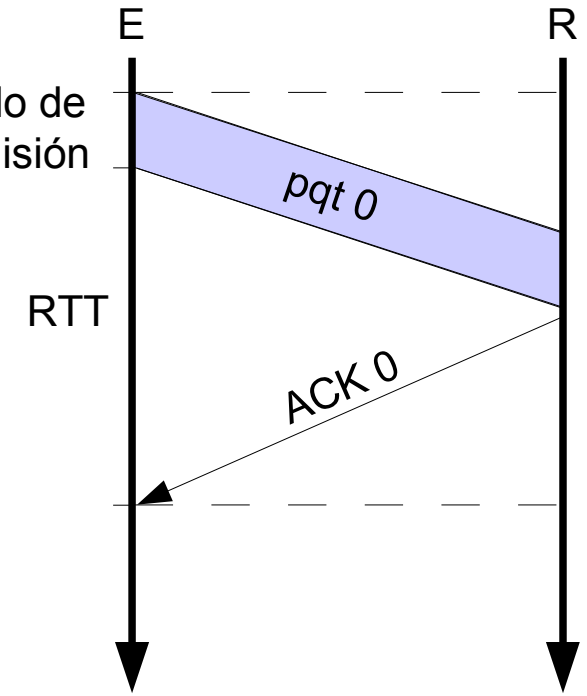
Distancia ida y vuelta entre E y R: 9000km

Velocidad de propagación: 300000 km/s

RTT: 9000 / 300000 = 30 ms

Total: 30.008 ms

Tasa de transmisión: 0.008 / 30.008 = 0.000267



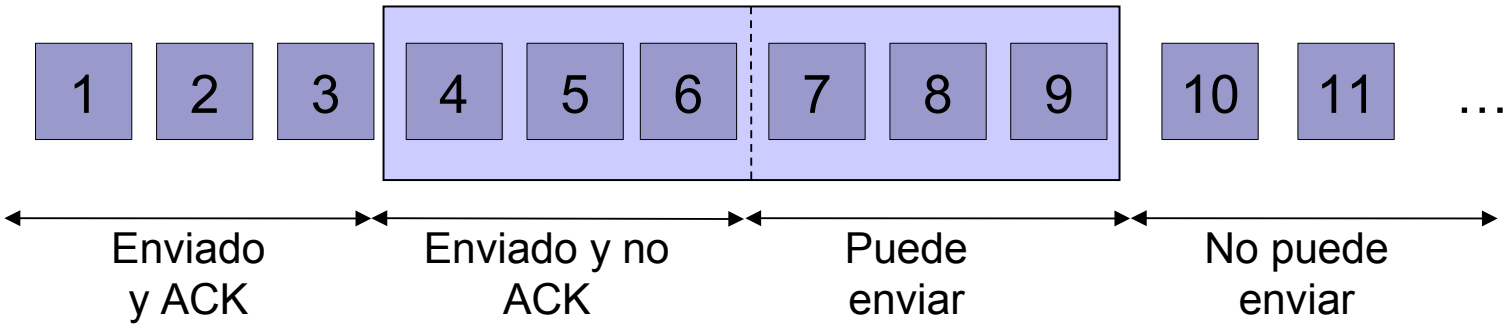
- ¡La velocidad de transmisión efectiva sólo es de 267 kbps!

Protocolos ARQ de parada y espera

- Solución: enviar varios paquetes sin esperar a los mensajes de confirmación → Procesamiento en cadena.
- ¿Qué necesito?
  - Aumentar el tamaño de los números de secuencia → Varios paquetes podrán estar en la red simultáneamente, sin confirmar.
  - El emisor necesitará un buffer para almacenar los paquetes transmitidos pero no confirmados.
  - El receptor necesitará un buffer para almacenar los paquetes recibidos correctamente (pero que la capa superior aún no puede procesar).
- Protocolos ARQ de procesamiento en cadena:
  - Retroceder N (GBN – Go-Back-N)
  - Repetición Selectiva (SR – Selective Repeat)

Protocolos ARQ retroceder N

- El emisor puede transmitir varios paquetes sin esperar a que estén confirmados.
- Se establece un máximo de N paquetes enviados sin confirmación.
- Se suelen representar los paquetes que se pueden enviar como una ventana que se van desplazando:
  - Cada vez que se recibe un ACK (nuevo) → Se puede enviar otro paquete.
  - Protocolo de **ventana deslizante**.
  - N = tamaño de ventana.
- Los números de secuencia son finitos → Son circulares.
- Sólo utiliza ACKs positivos, pero son **acumulativos**.
- http://media.pearsoncmg.com/aw/aw_kurose_network_4/applets/go-back-n/index.html



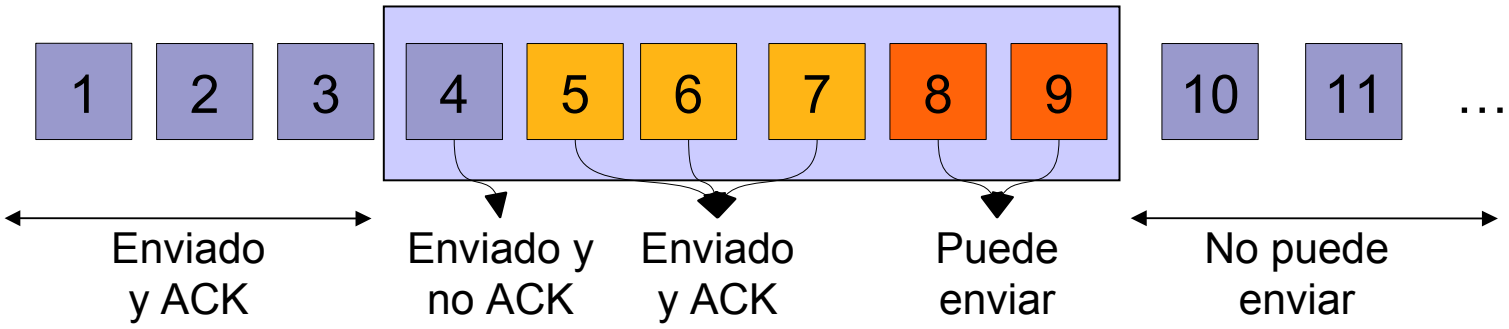
Protocolos ARQ retroceder N

- Funcionamiento del emisor:
  - Envía datos si tiene espacio en la ventana. Sino, espera.
  - Al recibir un ACK n, sabe que se han recibido correctamente todos los paquetes hasta n.
  - Si vence **el** temporizador de retransmisión → Envía **todos** los paquetes transmitidos y no ha recibido ACK.
- Funcionamiento del receptor:
  - Si recibe un paquete con número de secuencia n y está en orden (el anterior ha sido el n-1) → Se entregan los datos a la capa superior y envía un ACK n.
  - En el resto de casos, se descarta el paquete y se envía un ACK del último paquete recibido correctamente → No necesita buffer.

- El receptor no necesita buffer, aunque puede descartar paquetes recibidos correctamente.
- ¿Por qué no puedo enviar paquetes hasta infinito y ya iré recibiendo los ACKs?

Protocolos ARQ de repetición selectiva

- Problema de retroceder N: un error en un paquete hace que se repitan otros (muchos) paquetes recibidos correctamente.
- Solución: que el emisor únicamente retransmita los paquetes erróneos → Repetición selectiva.
  - Los ACKs son individuales (selectivos - SACK).
  - Se utiliza una ventana, pero con algunos paquetes confirmados.
  - El receptor necesita un buffer.
  - Se utiliza un temporizador para cada paquete enviado.
- http://media.pearsoncmg.com/aw/aw_kurose_network_4/applets/SR/index.html
- Corregido: http://www.eecis.udel.edu/~amer/450/TransportApplets/SR/SRindex.html



Protocolos ARQ de repetición selectiva

- Funcionamiento del emisor:
  - Si hay espacio en la ventana se envían datos, sino espera.
  - Si vence un temporizador → Se retransmite el paquete asociado.
  - Si se recibe un ACK →Se marca el paquete correspondiente como confirmado.
    - Si es el primero de la ventana, se desplaza hasta el primer paquete no confirmado.
- Funcionamiento del receptor:
  - Recibe un paquete de la ventana →Enviar ACK selectivo.
    - Si es nuevo → Almacenar en el buffer.
    - Si es el primero de la ventana, se entregan a la capa superior todos los paquetes hasta el primero sin confirmar y se desplaza la ventana.
  - Recibe un paquete anterior a la ventana → Enviar ACK selectivo.
  - En cualquier otro caso, ignorar el paquete.

Resumen

- Mecanismos para la transferencia fiable:
  - Temporizadores
  - Números de secuencia
  - Confirmación positiva (ACK)
  - Confirmación negativa (NAK)
  - Ventana

- ¿Qué pasa si el nivel inferior desordena los paquetes?
  - Con los números de secuencia puedo reordenar los paquetes.
  - Antes de enviar un número de secuencia *x*, debo estar seguro que un paquete *x* anterior no está circulando por la red → Se establece un tiempo de vida máximo (3 min para redes de alta velocidad).